Relational Consensus-Based Cooperative Task Allocation Management for IIoT-Health Networks

Carlos Pedroso[*], **Yan Uehara de Moraes**[*], **Michele Nogueira**[*†], **Aldri Santos**[*†]
[†]Department of Computer Science - UFMG, Brazil
[*]Department of Computer Science - UFPR, Brazil
Emails: {capjunior, yumoraes, michele, aldri}@inf.ufpr.br

***Abstract*—IIoT services focused on industry-oriented services often require objects run more than one task. IIoT objects poses the challenge of distributing and managing task allocation among them. The fairness of task allocation brings flexible network reconfiguration and maximizes the tasks to be performed. Although existing approaches optimize and manage the dynamics of objects, not all them consider both co-relationship between tasks and object capabilities and the distributed allocation over the cluster service. This paper introduces the ACADIA mechanism for task allocation in IIoT networks in order to distribute task among objects. It relies on relational consensus strategies to allocate tasks and similarity capabilities to determine which objects can play in accomplishing those tasks. Evaluation on NS-3 showed that ACADIA achieved 98% of allocated tasks in an IIoT-Health considering all scenarios, average more than 95% of clusters apt to performed tasks in a low response time, and achieved 50% more effectiveness in task allocation compared to the literature solution CONTASKI.**

***Index Terms*—Task Allocation, Cooperative Management, IIoT-Health.**

I. Introduction

The Internet of Things (IoT) is a heterogeneous network whose objects own characteristics like identity, physical attributes, computational and sensing capabilities [1], [2]. For the vast majority of IoT objects, it is important to reduce power consumption while communicating or making certain tasks. Meanwhile, objects must evenly share their resources and cooperate to support better network performance [3]. Thus, objects that run a set of functions can collaborate to allocate and realize different tasks [4]. In this context, Industrial Internet of Things (IIoT) has drawn greater attention [5] since it focuses on connecting multiple objects with numerous capabilities within an industrial environment, enabling thus every device to work in a synchronized and organized manner to perform sensing and monitoring tasks. Health application use the advanced technologies of IIoT to create interaction between patients and medical staff, hospital, and medical devices by creating a smart environment for the health domain [6], [7]. Reaching fairness performance in the distribution management of multiple tasks between IIoT objects is challenging due to configurations required by IIoT networks [8].

Inefficient allocations of sensing tasks between IIoT objects causes problems in distribution of computational resources, bad environmental setting, damaging the data collection and availability [9], [10]. In this way, we must preserve the efficiency of the allocation of tasks so that the objects are always able in emergencies to perform the reconfiguration of the environment and also deal with the entire volume of data generated. The wrong configuration of the environment makes it difficult to classify objects and makes the volume of data generated by them available since it takes a long time to reach the application and directly impacts its interpretation. In smart hospitals with high demand for multiple emergency services, the dynamic configuration between the environments and services must be done in an optimized, fast, distributed, and flawless manner [11]. Thus, intelligent management optimizes the resources of IIoT objects in the various tasks, by allowing objects to interleave execution according to the needs of the application and by preserving their resources [10].

Task allocation services have been extensively studied in WSN, usually treated as resource allocation for extending the network life [12], in IoT networks, that issue becomes essential for the arise of new applications [4]. Among the works that deal with the task allocation, many of them apply object virtualization in task groups [3] or distributed consensus [12]. Object virtualization applies assign tasks according to the sensing competencies of each object and its performance capacity in order to optimize the task execution to save resources. The distributed consensus applies to the equitable distribution of resources based on the interactions of each group of objects present in the network, this approach is applied in networks with a large number of participants [13].

However, these solutions disregard the similarity relationship between objects and the executing tasks. Besides that, they do not take into account an allocation decision based on the traits of the environment where objects are inserted. Hence, IIoT demands dynamic, adaptable, and fair solutions to handle a range of objects and to provide transparent configuration. Moreover, the solutions need to disseminate tasks among objects aware of their relationship with the environment and also the object capabilities [6] for getting balanced management of the IIoT network resource. Though, it is crucial for IIoT to provide mechanisms able to manage task allocation among objects by their relationships and capabilities for more robust and fairness management of available network resources.

This paper introduces ACADIA (*Rel**A**tional **C**onsensus-**BA**se**D** Task Allocation for **I**IoT-He**A**lth*) a mechanism for supporting sensing task allocation among objects into IIoT-Health networks. ACADIA arranges IIoT objects into similarity-based clusters to address an effective distribution of sensing tasks between available objects. This work extends to the health

978-3-903176-32-4

domain our previous work of task allocation in [14], as well as addressing the existence of multiple tasks. ACADIA employs collaborative relational consensus for better adaptation on the context, quick responses, and getting assertive decisions about the capabilities of objects to make specific sensing tasks. It also allows us to allocate simultaneously multiple and distinct tasks among the clusters. In analysis against CONTASKI [14] on the NS-3 simulator, ACADIA achieved 98% of suitably allocated tasks in a given IIoT-Health domain, average more than 95% of clusters apt to realize sensing tasks in a low response time and, achieved 50% more effectiveness in task allocation compared to CONTASKI.

This paper is organized as follows: Section II discusses the related work. Section III defines the model and assumptions taken by ACADIA. Section IV describes the ACADIA components and their operation. Section V shows the evaluation methodology to analyze the performance, and the results obtained. Section VI presents conclusions.

II. Related Work

The demand for dynamic and distributed services based on the resources and sensing capabilities of IoT objects has been the focus of several works [3], [12], [13], [15]. Though, most of them still face many issues by managing the allocation of the IoT resources, like co-relationship between tasks and object capabilities, fairness in multi-tasks distribution, and flexible network reconfiguration. In [3], an evolutionary algorithm based on heterogeneity recognition heuristic in IoT networks addresses to ensure greater stability and operational periods of tasks according to the current demands. The algorithm creates collaboration between the functions of objects by taking the task's demands and virtual objects groups selected to realize tasks and also reduce the energy consumption. In this solution, however, only a few objects are capable of performing the tasks attributed to network, as well as only two types of tasks can be done, which limits its use on networks including nodes with diverse capacities. In [12], virtual objects (VO) in an IoT smart health network realize the allocation of sensing tasks by a decentralized strategy, where VOs negotiate among them to reach a consensus on the resource allocation of the health devices. Despite it meets certain fairness in the task allocation, they employs only the same type of objects to perform the tasks, and hence ignoring the different sensing capacities, the varieties of interactions, and the impact of the network size.

In [14], we proposed the mechanism called CONTASKI for allocating task in IIoT that arranges the network into similarity-based groups to handle the division of tasks to be allocated. Although CONTASKI makes use of a distributed consensus strategy for decision making about the better task distribution for making a given service, we have ignored the simultaneous allocation of multiple distinct tasks, even being a condition expected in real networks. In [13], it is proposed a consensus-based heuristic approach to make decision on fault tolerance task allocation in IoT. The approach applies the concept of task groups and objects, so that in each task group, objects are chosen as virtual and vice-virtual. The model partly gets a flexibility on the network configuration, but it needs periodically to exchange *hello* messages, being computationally costly. In addition, they ignore the resources capabilities of nodes, which directly influence on the distribution of tasks. In [16], authors converted the task allocation issue in an IoT environment into an integration problem with a minimal degree variant to narrow the task allocation, and thus applying a genetic algorithm to reduce its execution time. Further, the objects only communicate each other by gateways services responsible for managing the interaction. However, the gateways restricts the relationships between nodes and can cause communication bottleneck depending on the network size. In [15], an algorithm decomposes sensing tasks in a sensor network into distributed ones, by taking the energy consumption of each task in order to get better resource allocation. They also apply a centralizing entity for distribution of roles, making it costly during the network reconfiguration. In [17], an algorithm for adaptive task mapping in sensors works jointly with the task scheduling based on a genetic algorithm to extend the network lifetime. Despite relating tasks and object capabilities, the centralization of distribution of tasks overloads the transmission channel and delays the message delivery, compromising the synchronization of tasks execution.

III. IIoT Health Environment

This section presents the structure of the IIoT health network, the manner how the objects communicate each other, and the model of sensing tasks in which ACADIA can run to realize the allocation management. We assume a hospital setting with multiple wings that can span over a number of floors and heterogeneous devices (IIoT objects) capable of making different classes of sensing relative to the building's environment and patients' physiological signals.

*1) **Network model**:* An IIoT network composed by a set of objects denoted by $N = \{ob_1, ob_2, ..., ob_n\}$ in an area (X_x, Y_y). All objects own an unique network identifier Id and differentiate each other by the sensing capabilities, represented by set $C = \{c_1, c_2, c_3, ., c_n\}$, processing power and memory. The objects are fixed in the setting and evenly distributed in the coverage area of the network. Also the objects do not suffer from energy restriction due to the existence of energy source in the network. Objects take roles as common or leader objects and Access Points (AP).

*2) **Communication model**:* Communication among the objects takes place over the wireless medium via a shared asynchronous channel, in which connections are reliable, and therefore the objects do not present communication failures. Also all objects exchange messages on the network layer. Further, the data sensed by objects can be accessed by the application layer regardless of the location, using protocols such as CoAP.

*3) **Sensing task model**:* Each task represents a demand for sensing the ambient and/or patients' physiological signals and it requires a set of sensing capabilities, whose size is variable, that depends on the setting ACADIA runs. Those tasks take place in a programmed manner with predefined time duration.

A task T is a tuple $\{T_{id}, C, \tau, q\}$, so that T_{id} is the task unique id; C means the set of sensing capabilities required to make the task, τ denotes the time needed to realize it and q the *per*-cluster quorum to perform the task. The sensing activity takes into account two classes: the infrastructural one senses the environment values such as temperature and light; the physiological one senses physiological signals like heartbeat and blood pressure as in [7]. The AP device keeps track of tasks' status pending, when the tasks are queued; dispatched, when they are accepted by some cluster, and completed.

IV. ACADIA

The ACADIA architecture comprises two modules, called **Cluster Coordination** (**CC**) and **Task Allocation Control** (**TAC**), as shown in Fig. 1. They act jointly to guarantee the configuration of clusters, as well as the dissemination and allocation of sensing tasks among IIoT-Health objects. The **CC** module arranges the network in virtual clusters and the **TAC** module controls the dissemination of multiple tasks to be done by the network objects according to their sensing capabilities. For achieving its goal, the modules exchange five types of messages: *CapabilityDissemination* that enable to configure the clustering; *LeaderRegister* that make the leaders to register into AP; *TaskDispatch* to dispatch tasks provided by the AP; *TaskAccept* that support leaders to accept tasks; *LeaderToCluster* that allow leaders to disseminate accepted task among the objects.

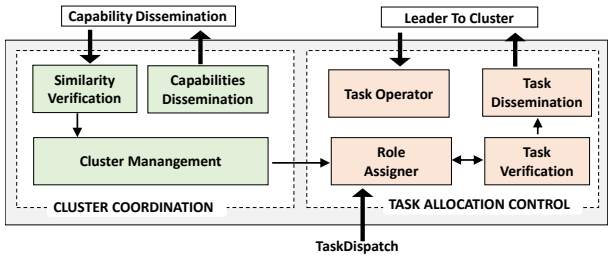


Fig. 1: The ACADIA Architecture

The **CC** module controls the creation and maintenance of the clusters by analyzing the neighboring objects using a similarity threshold of their capabilities in order to verify if they are apt to participate in the same cluster. Therefore, every time that CC receives a *CapabilityDissemination* message come from the neighbors it verifies such data about its identification, capabilities and number of neighbors. For that, **CC** account with three components: Capabilities Dissemination (CD), responsible for disseminating *CapabilityDissemination* messages with the object's identifier, its capabilities and number of neighbors; Similarity Verification (SV), which receives and verifies the fields of messages exchanged among the objects; and Cluster Management (CM), which manages the cluster creation using the objects' similarity, and the leader selection.

The **TAC** module coordinates the sensing task allocation according to the object capabilities and dispatches multiple tasks to the objects of the IIoT network, deviating from [14], in order to maximize and preserve their resources. It comprises the following components: Task Verification (TV), Role Assigner (RA), Task Dissemination (TD) and Task Operator (TO). TV oversees which tasks should be done and what capabilities are required by them. Whereas RA monitors the type of tasks to be assigned to the objects, employing relational consensus between the leader and tasks to evaluate which ones should be allocated according to cluster's capability. Next, DT dispatches the requested tasks considering the object capabilities. Lastly, TO takes care of the operation of the tasks came from the leader. Thus, the task allocation service becomes more fare and balanced, and does not overload the objects resources.

A. Cluster configuration

As the IIoT-Health size involves objects with different sensing capabilities, the **CC** module arranges the network objects in clusters based on leaders in order to create a network infrastructure capable to task allocate. Initially, objects begin cluster configuration exchanging *CapabilityDissemination* messages that carries the Id, capabilities and number of neighbors of the sender. Algorithm 1 describes the cluster configuration process. Initially, each object sends a *CapabilityDissemination* message in order to announce its Id, sensing capabilities ($MyCapabilities$) and number of neighbors ($NeighborhoodSize$), through the procedure `WarmUp`. When receiving a *CapabilityDissemination* message, the receiver updates its neighbors ($NeighList$), alongside with their capabilities ($NeighCapabilities$), by the procedure `RecvCapabilities`. The similarity verification takes into account those information, being calculated using cosine similarity. A neighbor can join into the cluster when its similarity is within the threshold. This update procedure occurs dynamically in all objects, ensuring that each one maintains its neighbor and cluster updated (procedure `SimilarityCalculation`).

The cluster leader selection (procedure `SelectLeader`) takes into account both the number of neighbors and individual capabilities to choose the leader. After that, the leader selected by the cluster informs to the AP that registers it as leader to guarantee the communication between AP, leaders and cluster members and a better hierarchical network organization.

Equation 1 computes the similarity value between two objects' capabilities, being based on [18, Eq. 3]. The similarity takes into account the object's own capabilities (C_{ob1}) and the neighbor's capabilities (C_{ob2}). In this division, the upper part calculates the norm of the vector that means the intersection between the capabilities. The bottom part takes the square root of the multiplication of the norm of each capability vector.

$$sim(ob_1, ob_2) = \frac{|C_{ob1} \cap C_{ob2}|}{\sqrt{|C_{ob1}| * |C_{ob2}|}} \quad (1)$$

The similarity value varies from 0 to 1, being that closer to 1, more similar two objects are, being that, the similarity level is labeled S_1 = Dissimilar, S_2 = Neutral and S_3 = Similar as a manner to show the levels of similarity objects and tasks get. This scale changes according to the previously established

Algorithm 1: Cluster configuration

```
1  procedure WarmUp
2      while 60 seconds has not elapsed do
3          BROADCASTCAPABILITIES()
4          RECEIVECAPABILITIES()
5      end
6  end procedure
7  procedure BroadcastCapabilities
8      Broadcast(MyId, MyCapacities, NeighborhoodSize)
9      WaitInterval()
10 end procedure
11 procedure RecvCapabilities
12     Id ← GetId()
13     NeightList ← NeighList ∪ Id
14     NeighCapabilities[Id] ← GetCapabilities()
15     NeighSize[Id] ← GetNeighborhoodSize()
16 end procedure
17 procedure SimilarityCalculation
18     foreach neighbor in NeighList do
19         NeighborCapabilities ← NeighCapabilities[neighbor]
20         sim = |MyCapacities ∩ NeighborCapabilities| / √(|MyCapacities| * |NeighborCapabilities|)
21         if sim ≥ Treshold then
22             cluster ← cluster ∪ neighbor
23         end
24     end
25 end procedure
26 procedure SelectLeader
27     LeaderCandidates ←
         GetNeighborsGreatestNeighborhoodSize(cluster)
28     Leader ←
         GetNeighborLargestCapabilitiesSet(LeaderCandidates)
29     if Leader == MyId then
30         SendLeaderRegisterToAP(MyId)
31     end
32 end procedure
```

capabilities before the IIoT is deployed and modifies according to the demand of application.

B. Task Allocation

Tasks are made available to carry out through the AP, which keeps a list of pending tasks and dispatches them according to settings' demands. The AP sends group leaders the tasks via $TaskDispatch$ messages. **TAC** plays on an IIoT-Health infrastructure established by the cluster configuration, and it runs guaranteeing resource maximization, i.e., allowing the task dissemination according to the capabilities of each cluster. Algorithm 2 describes the task allocation process and how leaders and the AP negotiate the tasks being performed. In order to identify the leaders, the AP monitors the *LeaderRegister* messages and keeps its leader list updated (procedure `APRecvLeaderRegister`). Initially, the AP manages a collection of pending tasks ($TaskList$). When dispatching tasks, the AP selects an amount (sm) of multiple pending tasks from the list and sends a *TaskDispatch* message to the cluster leaders announcing the task T to be executed (procedure `APSendTask`). After each dispatch, it waits for a time interval for receiving the confirmation (*TaskAccept* messages) from the available and compatible cluster leaders. When one confirmation is received at least, the AP removes it of the pending task list.

Once the leaders receive the task $T = (T_{id}, C, \tau, q)$, they verify the compatibility of their capabilities ($MyCapabilities$) with the capabilities C needed to perform the task, and if the number of objects in the cluster is greater or equal the quorum q needed. In case they meet the criteria, the cluster leader confirms with the AP (*TaskAccept* message) that it will perform the task and disseminates the task to the cluster. In case the cluster members cannot realize the task, the leader doesn't confirm this task with the AP (procedure `LeaderRecvTask`).

Algorithm 2: Task Allocation

```
1  procedure APRecvLeaderRegister
2      LeaderId ← GetId()
3      LeaderList ← LeaderList ∪ LeaderId
4  end procedure
5  procedure APSendTask(sm)
6      foreach dispatch round do
7          DispatchedTasks = DispatchMultipleTasks(sm)
8          if WaitConfirmation() then
9              TaskList ← TaskList − DispatchedTasks
10         end
11     end
12 end procedure
13 procedure LeaderRecvTask(T = (T_id, C, τ, q))
14     if C ⊆ MyCapabilities and |cluster| ≥ q then
15         SendTaskAccept(AP)
16         SendLeaderToCluster(T)
17     end
18 end procedure
```

C. Operation

ACADIA's task allocation acts dynamically and distributed in an IIoT-Health network on a hospital setting, whose objects are embedded in both medical equipment's and the structure of the building environment. The interactions between the IIoT objects occur over time and space dimensions, and objects in the transmission radius of the others exchange control messages in order to achieve a better configuration of the hospital activities. Fig. 2 illustrates how ACADIA acts for supporting the formation of IIoT objects cluster, leaders election, and allocation of tasks.The wireless signals mean objects within the transmission radius of each other and thus apt to exchange control messages about sensing capabilities. Each object carries its identifier Id and a sensing capabilities set C that it can make. Furthermore, capabilities c_1, c_2, c_3 and c_4 correspond to the sensing of temperature, humidity, lighting and body temperature, respectively. Besides, a similarity threshold ranging between 0 (weak) and 1 (strong) was setup for the formation of clusters, according to the capabilities of each object in the network.

We show the ACADIA operation on three different moments, said I_{t1}, I_{t2}, I_{t3} time instants. In I_{t1}, the set of objects (ob_1, ob_2, ob_3, ob_4) exchange messages about its sensing capability and neighborhood in order to realize the similarity calculation according to Eq. 1. The four objects obtain the following similarity values between them: $sim(ob_1, ob_2) = sim(ob_1, ob_4) = sim(ob_2, ob_4) = 1$ and $sim(ob_1, ob_3) = sim(ob_2, ob_3) = sim(ob_3, ob_4) = 0,87$. As the similarities

are within the range between S_2=Neutral and S_3=Similar, objects in that interval are clustered and it means objects with capabilities c_1, c_2, c_3.

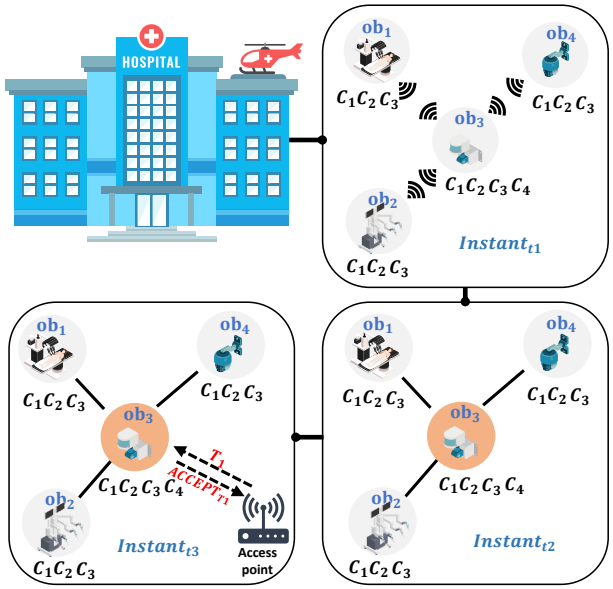


Fig. 2: Formation of clusters and distribution of tasks

In I_{t2}, ob_3 is elected the cluster leader because it has higher number of neighbors than the others. With the cluster coordinator operating in this way, each object keeps its neighborhood information and capabilities updated through the exchange of messages. Thus, objects in the spatial neighborhood are seen as members of the same cluster, in addition to ensuring better scalability to the network, since a hierarchy based on leaders aids in the quality of the information transferred. Moreover, it facilitates the distribution of tasks among the objects of the network. In I_{t3}, AP dispatches a task to the leaders, that verify whether their cluster is apt to do it. Over this view, ob_3, the leader, evaluates that the existing capabilities into the cluster are compatible and responds to the AP confirming that the cluster will realize such task. Other clusters might exist in different points in the network that are capable of carrying out other tasks sent along with task T_1.

V. Analysis

This section presents a performance evaluation of the ACADIA mechanism to assess its efficiency to the management of simultaneous tasks. We implemented ACADIA in NS3-simulator, version 3.29, and make all simulations taking into account an IIoT-Health scenario similar to a smart hospital with various levels. The object capabilities are classified in structural and health sensing functions. The former follows the ones in [19] that consist of *temperature*, *humidity*, *presence*, *light*, *position*, and *equipment condition*. The latter consists of *heartbeat monitoring*, *blood pressure*, *body temperature*, *oxygenation*, *glucose level* and *electrocardiogram* (ECG) [11]. We evaluate six IIoT-Heath scenarios in the same aforementioned hospital setting, with 50, 100 and 150 objects, and the number of simultaneous dispatched tasks, either 2 or 4 tasks, adding up to six scenarios configurations. The objects are evenly distributed in a area of *200 x 200* and communicate by IPv6 over IEEE 802.15.4. To avoid packet loss and bottleneck, we added a delay of *2ms* to messages exchanged among objects.

Scenarios with 2 dispatched tasks represent a configuration and sensing demand for an hospital wing with health services such as Emergency Response (ER), named "**Demand A (*D-A*)**". In contrast, scenarios with 4 dispatched tasks, the demand relates to comprehensive hospital services such as ER, Intensive Care Unit and Infirmary, named "**Demand B (*D-B*)**". We randomly generate each task and their duration employing *std::minstd_rand0* as random generator. Tasks' total duration span *1380s* to account for multiple tasks being dispatched. Seeds for each round constitute the sum of number of objects, run number and number of simultaneous dispatched tasks. *std::uniform_int_distribution* derive the objects capabilities and tasks using the random generator. The objects' capabilities, however, remained the same across the simulation rounds. Randomly producing those values brings variation to the simulation to account for real-life differences. If tasks required the same capabilities, given that objects do not have mobility and their capabilities remained fixed across simulation rounds, then the error would be close to zero, and, therefore, not representative. But, this randomization also reflects on higher standard deviation, leading to high amplitude error bars.

The system operates over *900s* and in the first *60s* there is an exchange of messages between all objects to disseminate their capabilities, followed by the similarity computation and leader register. The AP dispatches multiple tasks every *60s*, from *60s* to up *840s*, and clusters perform them for *60s* or *120s*, according to tasks' requirements. Moreover, each task has a random capabilities set, and all tasks require as structural capabilities at least *temperature*, *humidity* and *presence* and as health capabilities *heartbeat monitoring*, *blood pressure*, *body temperature*, *oxygenation*. The final capability set has up to three other structural capabilities and two other health capabilities, among the remaining ones of each type.

Pending tasks are forwarded to leaders by AP, always available, located in the center of the network, and equipped with a strong internet signal to reach all objects. The similarity parameter varies from 0 to 1. Clusters are classified as *apt* and *inapt* on each task dispatch, so that those *apt* can make the dispatched task, and ones considered *inapt* continue in an idle state saving resources for the next task. Also, considering the static scenario and transmission range of the objects, they form clusters with a non-deterministic number of participants. We also compare ACADIA with the CONTASKI system [14] to analyze both performance in the task allocation. We assess the two systems with the following metrics based on [3]: **number of clusters (NC)**, **Number of unallocated tasks (NUT)**, **number of allocated tasks (NAT)**, **clusters apt to perform tasks (CPT)**, **clusters inapt to perform tasks (CIT)**, **latency of task accept time (LAT)** and, **energy consumption (EC)**. All results correspond to the average of 35 simulations with a confidence interval of 95%.

A. Results

Fig. 3 (striped) shows the ACADIA performance for supporting the cluster formation, lighter striped bars represent the CPT and darker striped bars are the maximum. The CPT value relates to the similarity among objects that meets each capability set and the capabilities set of their neighbors, creating thus a consensual relationship between common objects and leaders to perform sensing tasks. ACADIA achieved an average CPT close to the average NC in most of the scenarios, showing that all clusters were apt to perform at least one of the dispatched tasks. In the scenario with 50 and 100 objects with either D-A and D-B the NC remained close to 4, and the CPT closely follows that average. The 150 objects scenario showed an NC of 6 for both demands. However, the CPT value had an average of 3.8 clusters, reaching up to 5 clusters. But, as it can be seen later, most of the tasks were performed. As for CONTASKI, it achieved higher CPT values, 4, 2.6 and 5.4, close to the averages of 5, 3 and 6. However, it did not translate in higher NAT values making explicit its inefficiency in task allocation.

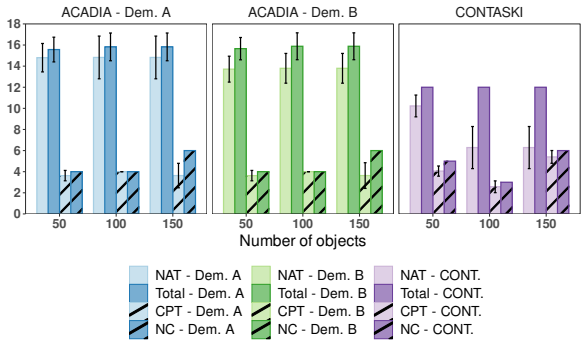
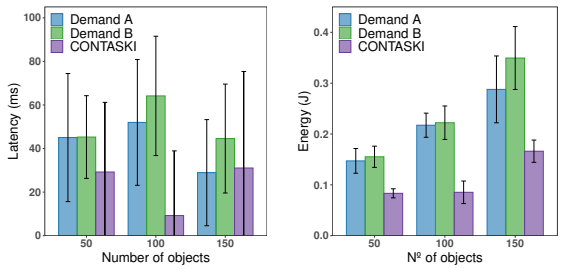


Fig. 3: Apt Clusters (CPT) & Task allocations (NAT)

Fig. 3 (non-striped) exhibit the amount of sensing tasks dispatched by AP. ACADIA's clusters were able to perform 98% of dispatched tasks on average, with some rounds performing 100% of them. But, ACADIA could not sustain 100% of allocated tasks in all simulation rounds due to the random capabilities assigned to each object. Scenarios with *D-A* had over 98% of allocated tasks and scenarios with *D-B* had over 86% of allocated tasks. In contrast, CONTASKI allocated 100% of tasks for the scenario with 50 objects and for others it only allocated 60% of them, even though it had higher CPT values, revealing, again, inefficient task allocation. ACADIA was able to allocate around 100% tasks, in all scenarios, obtaining 40% more of NAT than CONTASKI. Moreover, it is noteworthy that CONTASKI dispatches only one task, whereas ACADIA dispatches 2 and 4 tasks simultaneously.

Fig. 4a shows the graphs of the latency of task accept time that quantifies the difference between the task dispatch time and the last accept time as seen by the AP. ACADIA's *D-A* and *D-B*, with 50 objects, achieved similar LAT times around *45ms*, given they had the same CPT. The scenarios with 100 and 150 objects had the most variations between the demands due to factors mentioned previously. With 100 objects, *D-A* achieved *52ms* and *D-B*, *64ms*. In addition, with 150 objects LAT achieved *28ms* and *44ms*, respectively. Such variations observed are associated to the distance between objects and the AP, as well as the time taken by leaders to check it out if their capabilities are compatible to make the demand. Further, CONTASKI shows poor performance in LAT times, since its standard deviation is so high, the error bars reach values below zero (not showed in the graph). ACADIA, in its turn, achieved consistent LAT times in all scenarios. CONTASKI's LAT times are *30ms*, *10ms*, and *31ms*. ACADIA's higher LAT times – in the order of 50% higher than CONTASKI with 50 objects to almost four times with 100 objects – however, translated in effective task allocation, as showed in NAT analysis.



(a) LAT (b) EC

Fig. 4b shows the graphs about ACADIA's and CONTASKI's energy consumption for the task allocation management, only including energy spent on message exchange. It is noteworthy that ACADIA remained stable in energy consumption with little variation between the demands. ACADIA spent *0.15*, *0.22* and *0.29* J in scenarios with 50, 100 and 150 objects in *D-A*. The *D-B* spent *0.16*, *0.22* and *0.35* J, exhibiting less than 25% variation between the demands. While ACADIA exhibits a predictable trend in EC, CONTASKI displayed less energy consumption in all rounds. It spent *0.08*, *0.08* and *0.02* J exhibiting 80% less consumption then ACADIA's. However ACADIA's higher consumption translated close to 100% of tasks allocated in most rounds.

VI. Conclusion

This paper presented ACADIA for multiple tasks allocation on objects in an IIoT-Health network. It organizes the IIoT network into clusters based on the similarity of the capabilities of the objects and the neighboring objects. The mechanism applies the relational consensus to manage and distribute tasks between the clusters, considering the capabilities they inform. Results show the effectiveness of ACADIA in task allocation among IIoT objects. ACADIA was also compared to another task allocation mechanism, showing its effectiveness in multiple task allocation. As future work, we intend to evaluate scenarios with different types of mobility, priority in the task execution by the clustering and security concerns.

Acknowledgment

We would like to acknowledge the support of the Brazilian Agency CNPq, grants #436649/2018-7 and #426701/2018-6.

REFERENCES

[1] J. Gubbi, R. Buyya, S. Marusic, and M. Palaniswami, "Internet of things (iot): A vision, architectural elements, and future directions," *Future generation computer systems*, vol. 29, no. 7, pp. 1645–1660, 2013.

[2] A. Botta, W. De Donato, V. Persico, and A. Pescapé, "Integration of cloud computing and internet of things: a survey," *Future generation computer systems*, vol. 56, pp. 684–700, 2016.

[3] E. A. Khalil, S. Ozdemir, and A. A. Bara'a, "A new task allocation protocol for extending stability and operational periods in internet of things," *IEEE Internet of Things Journal*, 2019.

[4] Z. Ghanbari, N. J. Navimipour, M. Hosseinzadeh, and A. Darwesh, "Resource allocation mechanisms and approaches on the internet of things," *Cluster Computing*, pp. 1–30, 2019.

[5] L. D. Xu, E. L. Xu, and L. Li, "Industry 4.0: state of the art and future trends," *International Journal of Production Research*, vol. 56, no. 8, pp. 2941–2962, 2018.

[6] W. Qin, S. Chen, and M. Peng, "Recent advances in industrial internet: insights and challenges," *Digital Communications and Networks*, vol. 6, no. 1, pp. 1–13, 2020.

[7] J. J. P. C. Rodrigues, D. B. D. R. Segundo, H. A. Junqueira, M. H. Sabino, R. M. Prince, J. Al-Muhtadi, and V. H. C. D. Albuquerque, "Enabling technologies for the internet of health things," *IEEE Access*, vol. 6, pp. 13 129–13 141, 2018. [Online]. Available: https://doi.org/10.1109/access.2017.2789329

[8] M. Aazam, M. St-Hilaire, C.-H. Lung, I. Lambadaris, and E.-N. Huh, "Iot resource estimation challenges and modeling in fog," in *Fog Computing in the Internet of Things*. Springer, 2018, pp. 17–31.

[9] Y. Yuehong, Y. Zeng, X. Chen, and Y. Fan, "The internet of things in healthcare: An overview," *Journal of Industrial Information Integration*, vol. 1, pp. 3–13, 2016.

[10] M. Z. Hasan and H. Al-Rizzo, "Task scheduling in internet of things cloud environment using a robust particle swarm optimization," *Concur. and Computation: Pract. and Experience*, vol. 32, no. 2, p. e5442, 2020.

[11] A. M. Rahmani, T. N. Gia, B. Negash, A. Anzanpour, I. Azimi, M. Jiang, and P. Liljeberg, "Exploiting smart e-health gateways at the edge of healthcare internet-of-things: A fog computing approach," *Future Generation Computer Systems*, vol. 78, pp. 641–658, 2018.

[12] V. Pilloni and L. Atzori, "Consensus-based resource allocation among objects in the internet of things," *Annals of Telecommunications*, vol. 72, no. 7-8, pp. 415–429, 2017.

[13] G. Colistra, V. Pilloni, and L. Atzori, "Task allocation in group of nodes in the iot: A consensus approach," in *2014 IEEE International Conference on Communications (ICC)*. IEEE, 2014, pp. 3848–3853.

[14] C. Pedroso, Y. U. de Moraes, M. Nogueira, and A. Santos, "Managing consensus-based cooperative task allocation for IIoT networks," in *2020 IEEE ISCC*. IEEE, Jul. 2020. [Online]. Available: https://doi.org/10.1109/iscc50000.2020.9219635

[15] V. Pilloni and L. Atzori, "Deployment of distributed applications in wireless sensor networks," *Sensors*, vol. 11, no. 8, pp. 7395–7419, 2011.

[16] M. Kim and I.-Y. Ko, "An efficient resource allocation approach based on a genetic algorithm for composite services in iot environments," in *2015 IEEE International Conference on Web Services*. IEEE, 2015, pp. 543–550.

[17] Y. Jin, J. Jin, A. Gluhak, K. Moessner, and M. Palaniswami, "An intelligent task allocation scheme for multihop wireless networks," *IEEE Transactions on Parallel and Distributed Systems*, vol. 23, no. 3, pp. 444–451, 2011.

[18] O. B. Abderrahim, M. H. Elhedhili, and L. Saidane, "Ctms-siot: A context-based trust management system for the social internet of things," in *2017 13th IEEE IWCMC*, 2017, pp. 1903–1908.

[19] L. D. Xu and L. Duan, "Big data for cyber physical systems in industry 4.0: a survey," *Enterprise Information Systems*, vol. 13, no. 2, pp. 148–169, 2019.